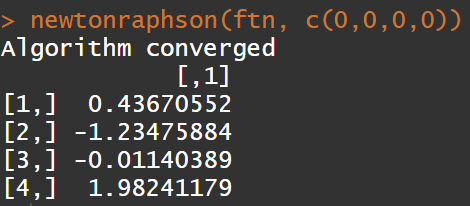
Ex 20-1：Data: resp.csv on ceiba, please read it into R and call it “resp” object：

1. Please use the Newton-Raphson method to find the maximum likelihood estimate (MLE) of the regression coefficients of logistic regression (). (4 points)

**Please note:** No need to make a plot for the initial value. Using the R built-in function “glm(outcome~treatment+age+baseline, family=binomial, data=resp)” to answer this homework will be scored as 0, although you may use it to check your own answer.



Code

newtonraphson <- function(ftn, x0, tol = 1e-9, max.iter = 100) {

x <- x0 # x0: the initial value

fx <- ftn(x)

iter <- 0

while ((max(abs(fx[[1]])) > tol) & (iter < max.iter)) {

x <- x - solve(fx[[2]]) %\*% fx[[1]]

fx <- ftn(x)

iter <- iter + 1

}

if (max(abs(fx[[1]])) > tol) {

cat('Algorithm failed to converge\n')

return(NULL)

} else { # max(abs(fx[[1]])) <= tol

cat("Algorithm converged\n")

return(x)

}

}

## 1

resp <- read.csv(file.choose(), header=T)

# change string levels into 1 and 0

resp$treatment <- ifelse(resp$treatment=="P", 1, 0)

X <- cbind(rep(1, length(resp$outcome)), resp$treatment, resp$age, resp$baseline)

dim(X)

Y <- resp$outcome

ftn <- function(betacoef){

pi1 <- exp(X %\*% betacoef) / (1 + exp(X %\*% betacoef))

gradient <- t(X) %\*% (Y - pi1)

hessian <- - t(X) %\*% diag(c(pi1 \* (1 - pi1)), length(resp$outcome)) %\*% X

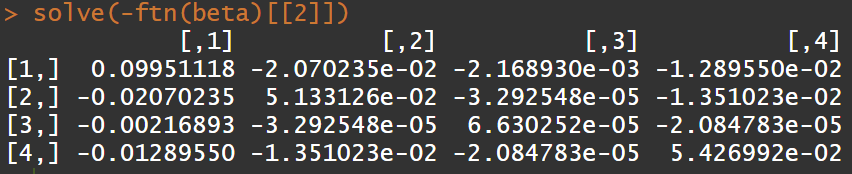
return(list(gradient, hessian))

}

newtonraphson(ftn, c(0,0,0,0))

1. Please find the variance-covariance matrix for . (2 points)

**Please note:** Using the R built-in function “vcov(model)” to answer this homework will be scored as 0, although you may use it to check your own answer.



Code:

beta <- newtonraphson(ftn, c(0,0,0,0))

beta

solve(-ftn(beta)[[2]])

1. Please find the log likelihood at . (2 points)

**Please note:** Using the R built-in function “logLik(model)” to answer this homework will be scored as 0, although you may use it to check your own answer.



Code:

ftn <- function(betacoef){

pi1 <- exp(X %\*% betacoef) / (1 + exp(X %\*% betacoef))

gradient <- t(X) %\*% (Y - pi1)

hessian <- - t(X) %\*% diag(c(pi1 \* (1 - pi1)), length(resp$outcome)) %\*% X

logLike <- sum(Y \* log(pi1 / (1 - pi1)) + log(1 - pi1))

return(list(gradient, hessian, logLike))

}

ftn(beta)[[3]]